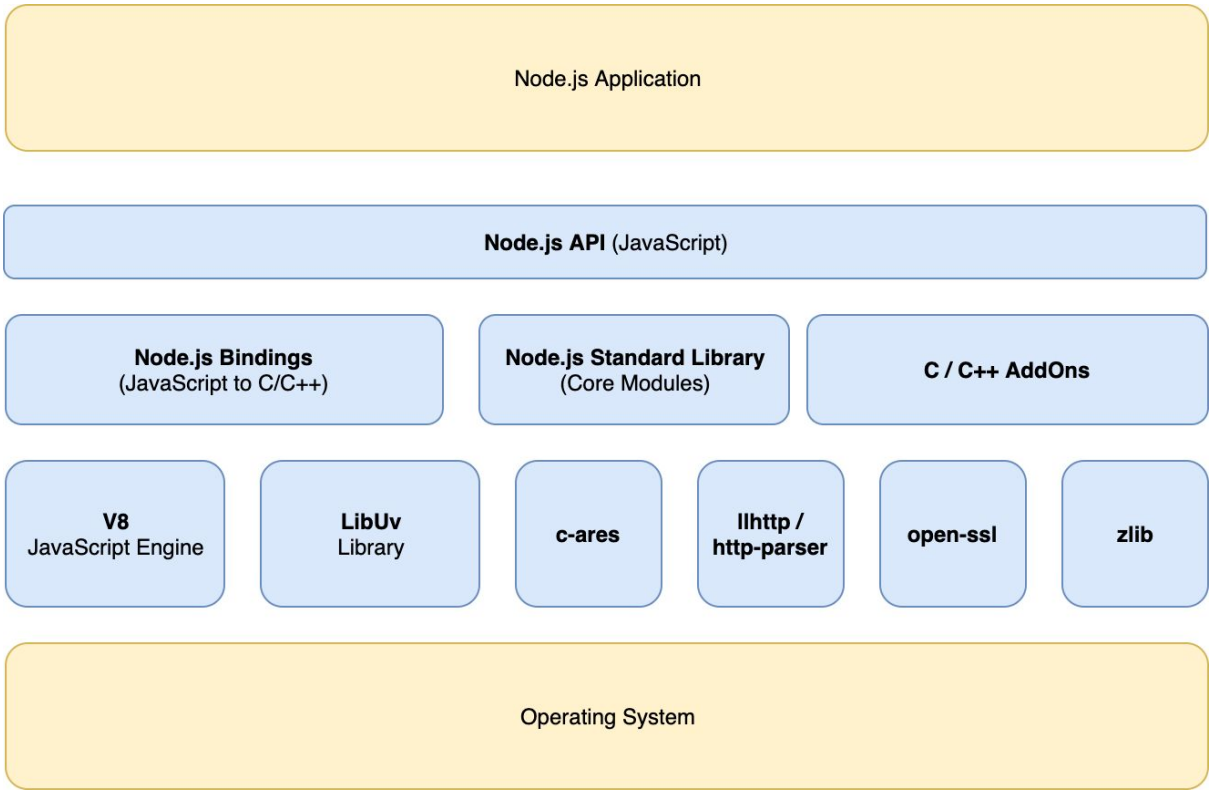
Node.js

Node.js[1] предсталява среда за изпълнение на JavaScript извън уеб браузър. Софтуерът е с неограничаващ отворен лиценз (MIT) и в момента (2020г.) е една от най-популярните технологии[4], използвани от софтуерните разработчици.

Node.js е най-пригоден за създаване на сървърни приложения. Отличава се със своя асинхронен модел, познат от JavaScript. Този модел се състои от прихващане на събития и последователна обработка на опашка от събития. Основното му техническо предимство пред други сървърни технологии е възможността за ефективна утилизация на хардуерните ресурси чрез неблокиране на основната нишка на приложението при входно-изходните операции, което води до подобрение на производителността и скалируемостта. Множество големи компании са включили Node.js към своя набор от технологии: Microsoft, Netflix, LinkedIn, PayPal, SAP, IBM и други.



Фиг. 1: Структура на слоевете на Node.js[3]

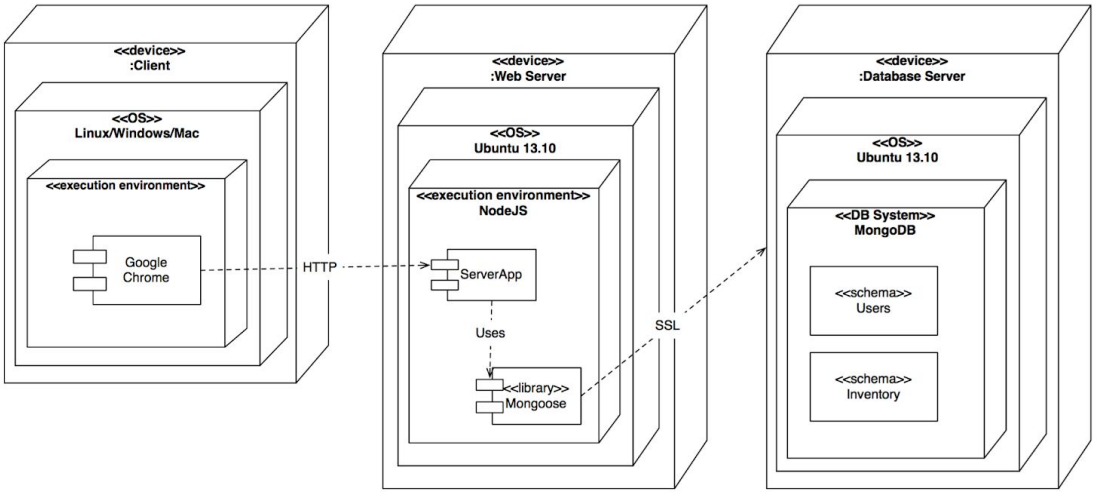
Структура, подходяща за представяне на архитектурата на Node.js е структурата на слоевете, представена на фиг. 1.

На най-горния слой е разположено Node.js API. Този слой представлява интерфейс, който Node.js приложенията използват за изграждане на бизнес логика. Node.js API е посредникът между приложенията и модулите на по-ниско ниво.

На по-долния слой се намира модул за комуникация на JavaScript код със C/C++, както и множество C/C++ библиотеки които могат да се използват в Node.js приложението с цел производителност. На същия слой се намира и Node.js стандартната библиотека, състояща се от основните модули.

На най-ниско ниво се намират модулите, които използват системните функции на операционната система:

- V8 е JavaScript engine-ът, който Node.js използва. V8 управлява memory heap, call stack, garbage collection и преобразуването на JavaScript код до машинни инструкции.
- LibUv е модулът, който осигурява асинхронната същност на Node.js. В него се намира thread pool, event loop и event queue.
- c-ares е модул, който осигурява възможност за асинхронни DNS заявки
- llhttp е модул за сериализация и десериализация на съобщения по HTTP протокол
- open-ssl предоставя криптографски функционалности
- zlib предоставя възможност за асинхронно компресиране и декомпресиране на данни



Фиг. 2: Структура на разположението[2]

Ролята на Node.js като компонент от системи, изградени по стил "клиент-сървър", може да се види чрез структурата на разположението, представена на фиг. 2. На нея се виждат три слоя - клиент, уеб сървър и сървър за база от данни. Node.js средата е разположена върху операционната система на уеб сървъра. В Node.js средата се намира сървърното приложение, което комуникира с клиента чрез HTTP протокол и използва библиотека за осъществяване на комуникация с базата от данни.

Като допълнителни структури могат да бъдат описани няколко структури на процесите:

- процес на обработка на събитие и процес на изпълнение на приложение с няколко асинхронни операции - с цел да внесат яснота за асинхронната същност на JavaScript, в частност Node.js
- процес на garbage collection - с цел да се добие представа как V8 engine-ът управлява паметта, за да се предприемат мерки за избягване на memory leak-ове, каквито не трябва да се случват в сървърни приложения

Допускам, че процесите, които предложих за представяне, може да са твърде сложни за представяне с по една UML диаграма. Смятам, че ще е необходимо да се направят по няколко диаграми за всеки процес, всяка представяща различна част от процеса и на различно ниво на абстракция.

Източници:

1. https://en.wikipedia.org/wiki/Node.js
2. https://rubenalcazar.gitbooks.io/nodejs-notes/nodejs_internals.html
3. https://medium.com/@chathuranga94/nodejs-architecture-concurrency-model-f71da5f53d1d
4. https://insights.stackoverflow.com/survey/2019/#technology-_-other-frameworks-libraries-and-tools